z5264786

COMP3331 assignment report

Program Design:

Language: Python 3.7.3

Message format:

- All messages are strings encoded using UTF-8
- Data is spaced out using spaces where this could pose an issue if encoded data contains any extra whitespace
- Upon being received they are decoded. And the data can be split using the split () function

How my program works:

1. Authentication
    a. Each account listed in credentials.txt has some number of attempts to successfully login before being blocked as input by the user when starting the server.
    b. The login attempt is counted if the input username is correct but password is incorrect
    c. If the same correct username but incorrect password combination is passed into the server multiple times the username is added to the block list for 10 seconds.
    d. In those 10 seconds if the user attempts to login with the blocked username an error message will be received
    e. There is a threading function that runs every second and checks each of the usernames in the block list and if username has been on the list for longer than 10 seconds it is removed.
    f. At this point the user can try to login to that previously blocked account
2. Multiple clients
    a. To handle multiple clients each time a client initialises a connection to the server. A thread is created to handle that client.
3. MSG command
    a. Program waits for an MSG command from the user
    b. Command creates a string with extra details like the message number, timestamp of the message, who sent the message and whether its been edited
    c. String is appended to the messagelog.txt file
    d. If message is sent the server will send an acknowledge message
4. RDM command
    a. Program waits for an RDM command from the user
    b. If messagelog.txt is empty an error message saying that there is no message history is sent back to client
    c. Else the server loops through the lines in the messagelog.txt file and converts all the times stamps to an epoch timestamp format. It then checks if the stored timestamp is greater than the input timestamp
    d. If stored timestamp is greater than the input timestamp the entire message is appended to a string

    e. The string is then sent to the client

5. EDT command
    a. Program waits for an EDT command from the user
    b. If messagelog.txt file is empty an error message saying there is no message history is sent back to the client
    c. Else server will loop through the messagelog.txt file and check if a message exists that matches the timestamp and message number. The server also checks if the user that sent the EDT command is also the user that sent the message.
    d. If any of the checks fail an error message is sent to the client
    e. To edit the messagelog.txt file as the program is looping through the lines of messagelog.txt all the lines are appended onto an array with the edited line if it exists to be appended instead of the original
    f. Then at the end of loop all the lines in the array are written to messagelog.txt
6. DLT command
    a. Program waits for an DLT command from the user
    b. If messagelog.txt file is empty an error message saying there is no message history is sent back to the client
    c. Else server will loop through the messagelog.txt file and check if a message exists that matches the timestamp and message number. The server also checks if the user that sent the DLT command is also the user sent the message.
    d. If any of the checks fail an error message is sent to the client
    e. To edit the messagelog.txt file as the program is looping through the lines of messagelog.txt all the lines are appended onto an array with the exception of the deleted line
    f. The message numbers are also adjusted using the length of the array
    g. After the loop all the lines are written to messagelog.txt
7. ATU command
    a. Program waits for an ATU command from the user
    b. If only one client is connected to server an error saying there is no other active users is sent back to the client
    c. Else server will loop through the userlog.txt file and append all lines to an array with an exception of the line contain details of the user that sent the ATU command
    d. The line is then formatted and sent to the client
8. OUT command
    a. The while loop operates on condition that loggedin = True
    b. The command sets loggedin to False thereby exiting the loop.

Design shortcomings / improvements:

- Timestamps
    - In processing any timestamps, they are first converted to an epoch format. For ease of comparison any input timestamps are also expected to be of epoch format. This makes many of the functionalities tedious to use and to test
- Lack of error cases for invalid user inputs. Program assumes correct inputs in most cases
- Lack of data persistence

    - For ease of implementation any data stored in any log files is wiped when server is closed or restarted
    - Adding persistent data would have been an interesting improvement that would have helped with testing
- Better code logic, the if-else statement has many inefficiencies that could only be solved with extensive refactoring
- OOP:
    - Code is very long and hard to debug sometimes. Following object orientated principles in refactoring the code would have helped with this

References:

Guide to socket programming in python:

https://realpython.com/python-sockets/

Handling of multiple clients:

https://codezup.com/socket-server-with-multiple-clients-model-multithreading-python/